

# VU Research Portal

## Between a hard and soft place: The (in)secure interplay of hardware and software

Cojocar, L.

2019

### **document version**

Publisher's PDF, also known as Version of record

[Link to publication in VU Research Portal](#)

### **citation for published version (APA)**

Cojocar, L. (2019). *Between a hard and soft place: The (in)secure interplay of hardware and software*. [PhD-Thesis - Research and graduation internal, Vrije Universiteit Amsterdam].

### **General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

### **Take down policy**

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

### **E-mail address:**

[vuresearchportal.ub@vu.nl](mailto:vuresearchportal.ub@vu.nl)



## Contents

<b>Acknowledgements</b>	<b>vii</b>
<b>Contents</b>	<b>viii</b>
<b>Publications</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Reverse engineering and systems' security . . . . .	2
1.2 Research questions . . . . .	4
1.3 Contribution and outline . . . . .	5
<b>2 Parser Identification in Embedded Systems</b>	<b>7</b>
2.1 Introduction . . . . .	7
2.2 Related work . . . . .	10
2.3 Static program analysis . . . . .	12
2.3.1 Identification of parser characteristics . . . . .	12
2.3.2 Lifting to an intermediate language . . . . .	13
2.3.3 Features of PARC <sub>3</sub> components . . . . .	14
2.4 Training and evaluation . . . . .	15
2.4.1 Scoring . . . . .	15
2.4.2 Validation . . . . .	16
2.4.3 Cross validation results . . . . .	16
2.5 Case studies . . . . .	19
2.5.1 GPS receiver . . . . .	19
2.5.2 Power meter . . . . .	19
2.5.3 Hard disk drive . . . . .	20
2.5.4 Programmable logic controller . . . . .	21
2.6 Future work . . . . .	24

2.7	Conclusion	25
<b>3</b>	<b>A Binary Solution for Switch-Case Recovery</b>	<b>27</b>
3.1	Introduction	27
3.2	The problem with patterns	29
3.2.1	Jump tables in practice	29
3.2.2	Pattern matching limitations	31
3.3	Tailored value set analysis for solving indirect jumps	33
3.4	Evaluation	36
3.4.1	Detailed analysis results	38
3.4.2	Comparing <i>JTR</i> with other solutions	39
3.5	Related work	42
3.6	Limitations	43
3.7	Conclusion	44
<b>4</b>	<b>Instruction Duplication</b>	<b>45</b>
4.1	Introduction	45
4.1.1	Motivation	46
4.1.2	Contribution	46
4.2	Background	47
4.3	Related work	48
4.4	Fault injection preliminaries	48
4.4.1	Fault injection background	49
4.4.2	Experimental fault injection setup	49
4.4.3	Fault injection characterization	50
4.5	Fault injection effectiveness	51
4.5.1	Inaccuracies in the fault injection model	51
4.5.2	Impact of compiler techniques	52
4.5.3	Case study: DFA attack on software AES-128	54
4.6	SCA of ID and infection countermeasures	56
4.6.1	Information-theoretic evaluation of ID for SCA	56
4.6.2	Converting infection to ID for SCA	57
4.7	Practical SCA results	61
4.7.1	Horizontal exploitation using CPA	61
4.7.2	Horizontal exploitation using templates	62
4.8	Conclusion	64
<b>5</b>	<b>ECC Memory and Rowhammer Attacks</b>	<b>65</b>
5.1	Introduction	65
5.2	Background	68
5.2.1	DRAM organization	68
5.2.2	Rowhammer	68
5.2.3	ECC in DRAM	69
5.3	Threat model	70
5.4	Summary of challenges	71
5.5	Challenge $C_1$ : reverse engineering ECC	72



5.5.1	Theoretical foundation . . . . .	72
5.5.2	Fault Injection . . . . .	73
5.5.3	Dealing with lack of syndromes . . . . .	75
5.5.4	Cold boot attacks . . . . .	76
5.5.5	Reverse engineering approach . . . . .	76
5.5.6	Results . . . . .	77
5.6	Challenge $C_2$ : ECC-aware Rowhammer . . . . .	85
5.6.1	Observing bit flips . . . . .	85
5.6.2	Controlling and composing bit flips . . . . .	88
5.7	Challenge $C_3$ : a practical <i>ECCploit</i> . . . . .	90
5.7.1	Templating correctable errors . . . . .	90
5.7.2	Combining bit flips . . . . .	91
5.7.3	Exploitation . . . . .	93
5.8	Related work . . . . .	95
5.9	Mitigations . . . . .	95
5.10	Conclusion . . . . .	96
<b>6</b>	<b>Conclusion</b>	<b>97</b>
	<b>Paper contribution</b>	<b>101</b>
	<b>References</b>	<b>103</b>
	<b>Summary</b>	<b>123</b>